Complexity of linear subsequences of Fibonacci-automatic sequences

Delaram Moradi[1], Narad Rampersad[2]
Jeffrey Shallit[1]

[1]David R. Cheriton School of Computer Science
University of Waterloo

[2]Department of Mathematics and Statistics
University of Winnipeg

Arithmetic operations using automata

- The arithmetic operations of addition and multiplication by a constant can be performed by a finite automaton reading its input in base-k.
- These operations can also be performed by an automaton that reads its input in the Fibonacci-base (Zeckendorf) numeration system.

State complexity of arithmetic operations

- How many states does such an automaton need to perform these operations?
- We will study this question for the Fibonacci-base.
- In companion work we look at integer bases.

Related work: multiples of n

- Charlier, R., Rigo, and Waxweiler (2011) showed that the minimal automaton accepting the language of multiples of n expressed in Fibonacci representation has exactly $2n^2$ states.
- Our problem is different: We want to find the size of an automaton accepting the language of Fibonacci representations of pairs $\{(i, ni) : i \geq 0\}$.
- It turns out that checking "divisibility by n" is somewhat different than checking the relation for "multiplication by n".

Automatic sequences

- Finite automata can also be used to compute infinite sequences.
- We consider sequences $(a(i))_{i\geq 0}$ computed by a finite automaton that reads i written in the Fibonacci-base and outputs $a(i)$.
- Given such a sequence, how many states are required for an automaton computing a shift $(a(i+c))_{i\geq 0}$?
- What about an arbitrary linear subsequence $(a(ni+c))_{i\geq 0}$?

A problem of Bosma and Don

- Bosma and Don studied the linear subsequences of the Fibonacci word

$$\mathbf{f} = 0100101001001010010100100101001010\cdots$$

generated by iterating the morphism $0 \to 01$, $1 \to 0$.

- They were interested in the size of the smallest morphism generating $(\mathbf{f}[ni + c])_{i \geq 0}$.
- We improve their exponential upper bound on the size of this morphism to polynomial.

Fibonacci representation

- Define the Fibonacci numbers, as usual, by $F_0 = 0$, $F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$ for $n \geq 2$.
- Fibonacci representation was introduced by Lekkerkerker (1952) and Zeckendorf (1972).
- Can also be obtained as a special case of the more general system of Ostrowski (1922).

Fibonacci representation

- Any n can be written as a sum of distinct Fibonacci numbers F_i for $i \geq 2$:

$$n = \sum_{2 \leq i \leq t} e_i F_i \text{ for } e_i \in \{0, 1\}.$$

- To ensure uniqueness, no two consecutive Fibonacci numbers can be used.
- A (most significant bit first) representation can be associated with the binary word of its coefficients

$$e_t e_{t-1} \cdots e_2.$$

Fibonacci representation

- We need a way to convert an arbitrary binary word $x = e_t \cdots e_2$ to the integer it represents.
- We write $[x]_F = \sum_{2 \leq i \leq t} e_i F_i$.
- For example, we have $[01001]_F = 6$.
- We allow leading zeros.

Representing pairs

- To represent pairs of numbers, we use the alphabet $\{0,1\} \times \{0,1\}$ and pad the shorter representation with leading zeros, if necessary.
- For example,

$$[0,1][0,0][1,1][0,0][1,0]$$

  represents the pair $(4, 11)$.
- If w and x have the same length then we write $w \times x$ to mean the word over $\{0,1\} \times \{0,1\}$ whose first component is w and whose second component is x.

Addition in Fibonacci-base

Theorem

For all integers $c \geq 0$, there exists an msd-first Fibonacci automaton M_c with at most $O(\log c)$ states accepting $x \times y$ where $[x]_F + c = [y]_F$.

Construction of M_c

- M_c reads input $x \times y$ in parallel and accepts if and only if $[y]_F = [x]_F + c$.
- We first define $D(x, y) = [y]_F - [x]_F$.
- We then find an interval I containing c such that if $D(x, y)$ ever falls outside I, then $D(x', y')$ stays outside I for all right extensions of x and y.
- A little analysis shows that we can take $I = [0, c]$.

Construction of M_c

$M_c = (Q, \Sigma_2, q_0, \delta, A)$ is constructed as follows.

- $Q \subseteq \{[a, b, d', d] \ : \ a, b \in \{0, 1\} \text{ and } d, d' \in I\}$,
- $q_0 = [0, 0, 0, 0]_F$,
- $\delta([a', b', d'', d'], [a, b]) = [a, b, d', d]$ where $a, b, a', b' \in \{0, 1\}, aa', bb' \neq 11$, and

$$d = d' + d'' + b' - a' + b - a,$$

- $A = \{[a, b, d', d] \ : \ a, b \in \{0, 1\} \text{ and } d = c\}$.

Transition rule of M_c

Consider processing the last two symbols of an input $xa'a \times yb'b$:

$$[a'', b'', d''', d''] \xrightarrow{[a',b']} [a', b', d'', d'] \xrightarrow{[a,b]} [a, b, d', d].$$

Then

- $[x00]_F = [x]_F + [x0]_F$ (Fibonacci recurrence), so
- $[xa'a]_F = [x]_F + [x0]_F + 2a' + a$, and hence
- $D(xa'a, yb'b) = D(x, y) + D(xa', yb') + b' - a' + b - a$.

Specifying the state set Q

- Define $d' = D(x, y)$, $d = D(xa, yb)$.
- Let $\alpha = (1+\sqrt{5})/2$ and $\beta = (1-\sqrt{5})/2$, the zeros of the polynomial $X^2 - X - 1$. Then

$$-\beta^2 < [x0]_F - \alpha[x]_F < -\beta.$$

- Using this we get three cases for d':
  - *Case 1*: $a = b$. Then $d/\alpha - 1 < d' < d/\alpha + 1$.
  - *Case 2*: $a = 0$, $b = 1$. Then $d/\alpha - 2 < d' < d/\alpha$.
  - *Case 3*: $a = 1$, $b = 0$. Then $d/\alpha < d' < d/\alpha + 2$.

Specifying the state set Q

- Starting from an accepting state $[a, b, d', c]$, we determine the possible predecessors of this state, and the possible predecessors of those, etc.
- We group the predecessors together in levels.
- The states of the form $[a, b, d', c]$ are at level 0, and the predecessors of these are at level 1, and so forth.

Counting the states per level

- States in level i are of the form $[a, b, d', d]$ and $d \in [D_i, D_i + r_i]$ for certain D_i, r_i.
- $D_0 = c$ and $r_0 = 0$.
- Smallest d' at level i is $\lfloor D_i/\alpha \rfloor - 1$
- Largest d' is $\lceil (D_i + r_i)/\alpha \rceil + 1$
- Hence $r_{i+1} < r_i/\alpha + 4$.
- Starting with $r_0 = 0$, using this inequality four times shows that $r_i \leq 8$, and hence there are at most 9 possible values of d at each level.

Counting the number of levels

- We have $d' < (d+2)/\alpha$ and thus

$$D_{i+1} + r_{i+1} < (D_i + 10)/\alpha < D_i/1.1$$

  provided $D_i \geq 22$.
- There are at most $\log_{1.1} c = O(\log c)$ levels until $D_i \leq 22$.
- When $D_i \leq 22$, we just include all states $[a, b, d', d]$ with $d \leq 22$ (a constant number).
- Hence there are only $O(\log c)$ co-accessible states.

Multiplication in Fibonacci-base

Theorem

For all integers $c \geq 0$, there exists an msd-first Fibonacci automaton with at most $O(n^2)$ states accepting $x \times y$ where $n * [x]_F + c = [y]_F$.

Fibonacci-automatic sequences

- A Fibonacci-DFAO $M = (Q, \{0, 1\}, \Delta, \delta, q_0, \tau)$ is a DFA with an output function $\tau : Q \to \Delta$ instead of accepting states.
- It generates a Fibonacci-automatic sequence $(h(i))_{i \geq 0}$ in the following way:
- The input is an msd-first representation x of i in the Fibonacci-base, and the output $h(i)$ is $\tau(\delta(q_0, x))$.

Properties of a Fibonacci-DFAO

- words with input 11 do not lead anywhere, or lead to a dead state, at the cost of 1 extra state;
- a word with in-arrow labeled 1 has an out-arrow only on 0;
- words with in-arrow labeled 0 has arrows out on both 0 and 1;
- inputs with leading zeros always give the same output

The Fibonacci word

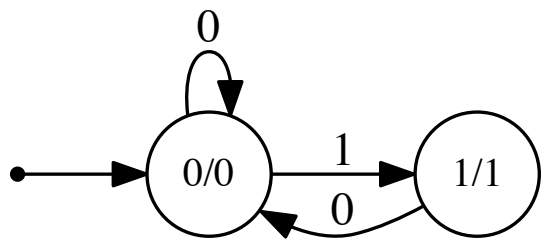
The famous infinite Fibonacci word

$$\mathbf{f} = 01001010\cdots,$$

defined as the fixed point of the morphism

$$0 \longrightarrow 01, \quad 1 \longrightarrow 0$$

can be generated by the following Fibonacci-DFAO:



Shifts of a Fibonacci-automatic sequence

Theorem

Let $(h(i))_{i\geq 0}$ be a Fibonacci-automatic sequence generated by a DFAO of m states. For an integer constant $c \geq 0$, there is a DFAO of $O(m^2c^2)$ states generating $(h(i+c))_{i\geq 0}$.

The interior sequence

- Let $(h(i))_{i\geq 0}$ be an automatic sequence with minimal DFAO $(Q, \Sigma, \Delta, \delta, q_0, \tau)$.
- There is a unique associated interior sequence $(h'(i))_{i\geq 0}$ taking its values in Q, whose DFAO is obtained by replacing $\tau : Q \to \Delta$ by the identity map on Q.
- Thus $(h(i))_{i\geq 0}$ is the image of $(h'(i))_{i\geq 0}$ under the coding $q \to \tau(q)$.

The automaton for $(h(i+c))_{i\geq 0}$

We define an automaton M' that on input x reaches a state of the form

$$[[h'([x]_F), a'_0], \ldots, [h'([x]_F + c), a'_c]],$$

where a'_i is the last letter in $([x]_F + i)$, and after reading the letter a transitions to the next state

$$[[h'([xa]_F), a_0], \ldots, [h'([xa]_F + c), a_c]].$$

The automaton for $(h(i+c))_{i\geq 0}$

- The states of M' are lists of pairs

$$[[p_0, a_0], \ldots, [p_c, a_c]]$$

where $p_i \in Q$ and $a_i \in \Sigma_2$.

- For a state $[[h'([x]_F), a'_0], \ldots, [h'([x]_F + c), a'_c]]$, the first components of the pairs spell out the length-$(c+1)$ factor of h' from position $[x]_F$ to position $[x]_F + c$.

The automaton for $(h(i+c))_{i\geq 0}$

- The second components of $[[h'([x]_F), a'_0], \ldots, [h'([x]_F + c), a'_c]]$, spell out a length-$(c+1)$ factor of the Fibonacci word **f**!
- a'_i is the last letter in $([x]_F + i)$, but this is exactly equal to $\mathbf{f}[[x]_F + i]$.
- The output function is $\tau'([[p_0, a_0], \ldots, [p_c, a_c]]) = \tau(p_c)$.

Number of states in M'

- $p_0, \ldots, p_c$ is a length-$(c+1)$ factor of $(h'(i))_{i \geq 0}$.
- How many such factors can there be in h'?
- In general, a Fibonacci sequence generated by a DFAO with m states has $O(m^2 n)$ factors of length n.
- So there are $O(m^2 c)$ possibilities for $p_0, \ldots, p_c$.
- The $a_0, \ldots, a_c$ form a length-$(c+1)$ factor of $\mathbf{f}$, and it is well-known that there are exactly $c+2$ such factors.
- So the automaton M' for $(h(i+c))_{i \geq 0}$ has $O(m^2 c^2)$ states.

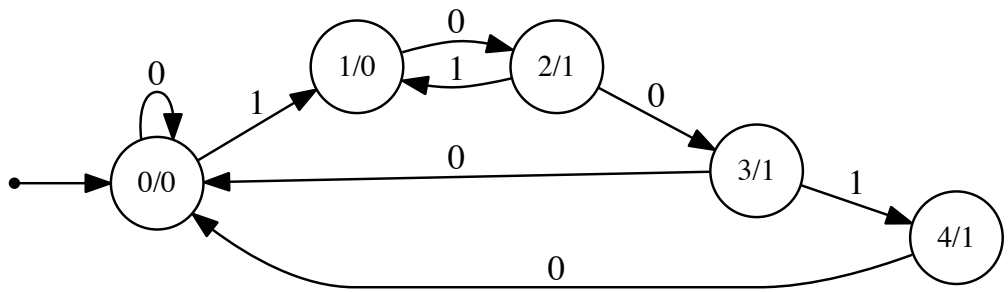
Linear subsequences

Theorem

Let $n \geq 1$ and $0 \leq c < n$, and let $(h(i))_{i \geq 0}$ be a Fibonacci-automatic sequence generated by a DFAO of m states. There is a DFAO of $O(m^2 n^4)$ states recognizing the linear subsequence $(h(ni + c))_{i \geq 0}$.

An example for $\mathbf{f}[2i]$

The minimal DFAO for $\mathbf{f}[2i]$ is



Non-tightness of the bound

- We do not know examples where the n^4 bound is actually attained.
- Perhaps a tight upper bound is $O(n^2)$.
- For $c = 0$ and $h(i) = \mathbf{f}[i]$, the infinite Fibonacci word, the number of states in the smallest DFAO for $\mathbf{f}[ni]$ is OEIS sequence `A385021` (calculated with `Walnut`).
- The first 10 terms are

$$2, 5, 10, 17, 27, 36, 52, 65, 78, 103.$$

Converting between DFAO's and morphisms

States of the Fibonacci-DFAO become letters of a morphism h. The substitution rules of the morphism are of the form

$$p \to qr$$

if $p \xrightarrow{0} q$ and $p \xrightarrow{1} r$ are transitions in the DFAO or just

$$p \to q$$

if there is no transition on 1 from state p.

Converting between DFAO's and morphisms

- We also have a coding g that is identical to the output function of the Fibonacci-DFAO.
- Then the morphic sequence $g(h^{\omega}(q_0))$ gives the original Fibonacci-automatic sequence.

Converting the $\mathbf{f}[2i]$ DFAO

If we convert the automaton for $(\mathbf{f}[2i])_{i\geq 0}$ to a morphism γ and coding τ we get:

$$\gamma : 0 \to 01, \quad 1 \to 2, \quad 2 \to 31, \quad 3 \to 04, \quad 4 \to 0$$

$$\tau : 0, 1 \to 0, \quad 2, 3, 4 \to 1.$$

When we iterate the morphism γ we get the sequence $01231040\cdots$ and after applying the coding τ we get $00110000\cdots$.

Bosma and Don's bound

- This morphism was used as an example by Bosma and Don (2024).
- They proved an exponential (in n) upper bound on the size of a morphism for $(\mathbf{f}[ni])_{i\geq 0}$.
- Our results, along with the conversion from DFAO to morphism, improve this to polynomial.

Bound for morphism size

Corollary

Let $n \geq 1$ and $0 \leq c < n$, and let $(h(i))_{i \geq 0}$ be a Fibonacci-automatic sequence generated by a DFAO of m states. There is a morphism of size $O(m^2 n^4)$ generating the subsequence $(h(ni + c))_{i \geq 0}$.

Büchi arithmetic

- We can also study these state complexity results from another point of view.
- Instead of trying to directly construct the automaton for a given arithmetic operation, we analyze the size obtained when constructing it using the primitives available in an implementation of Büchi arithmetic, such as Walnut.
- For details, see the paper.

Other related results

Surprisingly, for shifts of the Fibonacci word there is a much stronger result.

Theorem (Moradi, Popoli, Shallit, and Vukusic (2026))

Let $(f(i))_{i\geq 0}$ be the Fibonacci word $\mathbf{f}$. For $c \geq 0$, there exists an lsd-first DFAO generating $(f(i+c))_{i\geq 0}$ with $O(\log c)$ states. The same holds for msd-first representation.

Open Questions

- What is the tight bound for the state complexity of $(h(ni+c))_{i\geq 0}$?
- What about other numeration systems? Tribonacci base?

The End